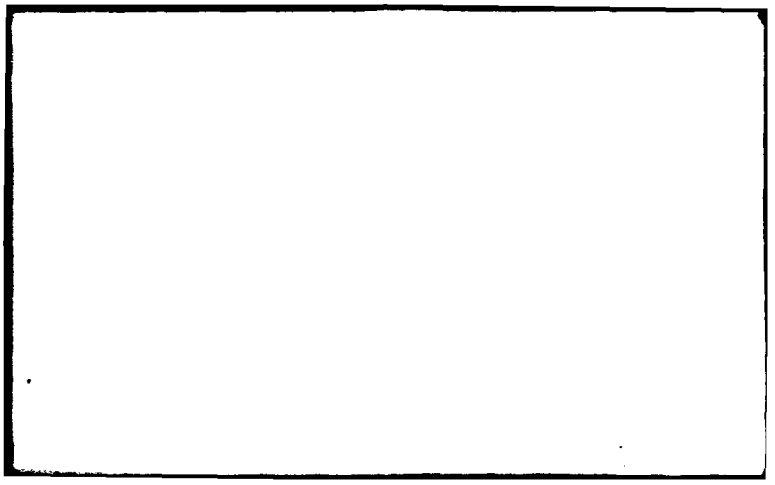
AD-A095 657 HARVARD UNIV CAMBRIDGE MASS AIKEN COMPUTATION LAB F/G 9/2
ON PROBABILISTIC AND SYMMETRIC PARALLEL COMPUTATIONS,(U)
1980 J H REIF NSF-MCS79-21024
UNCLASSIFIED TR-22-80 NL

END
DATE FILMED
3 81
DTIC



(2)

ON PROBABILISTIC
AND
SYMMETRIC PARALLEL COMPUTATIONS

BY

John H. Reif

TR-22-80

1980

(6)

On Probabilistic and Symmetric Parallel Computations

(12) 15

by

(10) John H. Reif

(11) 1980

(14) TR-11-80

Aiken Computation Lab., Harvard University, Cambridge, MA 02138
This work was supported in part by the National Science ~~Foundation Grant~~ NSF-MCS79-21024 and the Office of Naval Research Grant N00014-80-C-0647.

(15) NSF-MCS79-21024

I. Introduction

There are many known examples of problems for which there is a probabilistic (sequential) algorithm more efficient than any known deterministic algorithm. For example primality testing (see [Miller, 75], [Rabin, 74], and [Strassen and Solovay, 74]).

This paper considers the question: Are parallel machines made more efficient by allowing processors to take probabilistic choices?

We also consider the question: What is the computational power of parallel machines, if we require transitions of processors to be symmetric?

This is motivated by physical constraints to parallel computation such as energy consumption (see [Toffoli, 79]). Also, there appears to be some fundamental relationship between symmetrical and probabilistic computations.

To partially answer these questions, we utilize previous studies of the computational complexity of parallel machines (see references below), studies of probabilistic computations by [Adleman and Manders, 77], [Adleman, 78], and [Gill, 77], and work on symmetric computations by [Lewis and Papadimitriou, 80], and work relating probabilistic and symmetric computations [Aleliunas, Karp, Lipton, Lovász, Rackoff, 79].

We formulate a general thesis for how both the probabilistic and symmetric types of parallel computation relate to other types of computation.

Probabilistic Parallel Processing Thesis

(1) probabilistic // time $T(n)$

$\subseteq$ deterministic time $T(n)^{O(1)}$/advice $T(n)^{O(1)}$

(2) probabilistic // space $S(n)$

$=$ // space $S(n)^{O(1)}$

Symmetric Parallel Processing Thesis

(3) symmetric // time $T(n)$

$=$ // time $T(n)^{O(1)}$

(4) symmetric // space $S(n)$

$\subseteq$ probabilistic // simultaneous space $S(n)^{O(1)}$, time $2^{O(T(n))}$.

We provide evidence for the above by proving them to hold for a wide class of parallel machine models, including: alternating TMs [Chandra, Kozen, Stockmeyer, 78], parallel RAMs (Fortune and Wyllie, 78], [Savitch and Stimson, 78], and hardware modification machines [Dymond and Cook, 80].

Note that our approach is similar to that of [Goldschlager, 78] who provided evidence for the

parallel processing thesis

$$\text{// time } T(n) = \text{deterministic space } T(n)^{O(1)}.$$

Recent work by [Hong, 80] proposed to generalize the parallel processing thesis to *all computational types*; but our results for probabilistic and symmetric parallel machines appear to directly contradict his generalization (we view this as a significant aspect of our complexity results).

(For brevity and improved readibility of this draft, we have deleted many interesting simultaneous complexity relations and emphasized simply time and space complexities.)

2. Definitions

2.1 Nondeterministic Automata

Let a *nondeterministic automaton* (a N-automaton) M consist of

(i) a finite *input alphabet* Σ,

(ii) a set $\mathcal{I}$ of *instantaneous descriptions* (IDs)

(iii) an *initialization function* $INIT: \Sigma^* \to \mathcal{I}$

(iv) a set $ACCEPT \subseteq \mathcal{I}$ of *accepting* IDs,

(v) a *transition relation* $\vdash \subseteq \mathcal{I} \times \mathcal{I}$ with the restriction:

$$\exists d > 0 \quad \forall I \in \mathcal{I} \quad (|NEXT(I)| \leq d)$$

where $NEXT(I) = \{J / I \vdash J\}$.

We also assume $NEXT(I) = \emptyset$ for all $I \in ACCEPT$.

Let $G = (\mathcal{I}, \vdash)$ be the *computation graph* of M, with node set $\mathcal{I}$ and edges given by $\vdash$. Let a *computation sequence* be a nonempty maximal path in G. Let a computation sequence $I_0, \ldots, I_k$ be *accepting* if it is finite, and $I_k \in ACCEPT$. For any input $w \in \Sigma^*$, let M *accept* w if there is an accepting computation sequence starting at INIT(w).

Let $\vdash$ be *deterministic* if

$$\forall I \in \mathcal{I} \quad (|NEXT(I)| \leq 1)$$

and *symmetric* if

$$\forall I, J \in \mathcal{I} \; (I \vdash J \text{ iff } J \vdash I).$$

M is *deterministic* (a D-automaton) if $\vdash$ is.

M is *symmetric* (a SYM-automaton) if $\vdash$ is.

(Note. SYM-TMs were first defined by [Lewis and Papadimitriou, 80]; this definition requires a technical augmentation of the state transition function of a TM to allow symmetry between a transition and its reverse.)

2.2 Probabilistic Automata

Let us consider each computation sequence $I_0, I_1, \ldots$ to be generated by the following probabilistic process: for $i = 0,1,\ldots$

Let I_{i+1} be a randomly chosen element of $\text{NEXT}(I_i)$. M is *probabilistic* (an R-automaton) if $\forall I \in \mathcal{I}$, if R has at least one accepting computation sequence starting at I, then any randomly generated computation sequence from I is accepting, with probability $\geq 1/2$.

(Note if the basic automaton is a TM, then these are essentially R-TMs, as defined by [Adleman, 78].)

2.3 Alternating Automata

Let M be *alternating* (an A-automaton) if addition to (i)-(v) above,

(vi) $\mathcal{I}$ is partitioned into sets $\mathcal{I}_U$ (the *universal* IDs) and $\mathcal{I}_E$ (the *existential* IDs).

A *computation tree* of M is a tree T with each node v labelled $\ell(v) \in \mathcal{I}$ such that

(a) if $\ell(v) \in \mathcal{I}_U - \text{ACCEPT}$, then $\text{NEXT}(\ell(v))$ are precisely the labels of the siblings of v.

(b) if $\ell(v) \in \mathcal{I}_E - \text{ACCEPT}$, then v has at least one sibling labelled from $\text{NEXT}(\ell(v))$.

The computation tree T is *accepting* if it is finite and its leaves are labelled from ACCEPT. Given input $w \in \Sigma^*$, M *accepts* w if there is an accepting computation tree with root labelled with $\text{INIT}(w)$.

For $X,Y \in \{U,E\}$, let

$$\vdash_{XY} \; = \; \vdash \cap \; (\mathcal{I}_X \times \mathcal{I}_Y) \; .$$

Let M be *symmetric alternating* (a SYM-A-automaton) if $\vdash_{UE}$, $\vdash_{EU}$ are deterministic, and $\vdash_{EE}$, $\vdash_{UU}$ are symmetric.

It is possible to view an A-automaton as a *game* (see [Chandra, Kozen, Stockmeyer, 78]), where the E-player attempts to make transitions eventually yielding an ID in ACCEPT, irregardless of the possible transitions by the U-player. Thus a SYM-A-automaton is a *symmetric game*, where each of the E and U player's transitions are symmetric. Note that if we

were to allow $\vdash_{EU}$ or $\vdash_{UE}$ to be symmetric, then the game would be trivial. An example of a symmetric game is a pursuit game on a undirected graph.

2.4 Probabilistic Alternating Automata

Let M be an A-automaton. For $X \in \{U,E\}$, let a computation sequence $I_0, I_1, \ldots$ be of *type* X if each transition is a $\vdash_{XX}$ transition.

For any $I \in \mathcal{I}$, let P_I be the probability that a randomly generated computation sequence of the same type as I (i.e., of type E if $I \in \mathcal{I}_E$ and of type U else) ends at an ID J such that there is an accepting computation tree with root labelled J.

Let M be *probabilistic alternating* (a RA-automaton if

(a) $\forall I \in \mathcal{I}_E$, $P_I > 0 \Rightarrow P_I \geq 1/2$

(b) $\forall I \in \mathcal{I}_U$, $P_I < 1 \Rightarrow P_I < 1/2$.

2.5 Complexity Classes

Let $\mathcal{M}$ be a collection of automaton; with complexity measures time and space.

Let $\mathcal{M}\text{-TIME}(T(n))$ = {L/L is accepted by $M \in \mathcal{M}$ in time T(n)}.

Let $\mathcal{M}\text{-SPACE}(S(n))$ = {L/L is accepted by $M \in \mathcal{M}$ in space S(n)}.

We also define a simultaneous complexity class related to Steve Cook's Class: for $d \geq 0$, let $\mathcal{M}\text{-SC}^d(S(n))$ = {L/L is accepted by $M \in \mathcal{M}$ in simultaneous space $S(n)^d$ and time $2^{O(S(n))}$}.

We use the notation:

$$L(\mathcal{M}) = \{L/L \text{ is accepted by } M \in \mathcal{M}\}$$

$$coL(\mathcal{M}) = \{\Sigma_M^* - L/L \text{ is accepted by } M \in \mathcal{M}\}$$

Also, we use the notation:

$$L(\mathcal{M})/f(n) \text{ advice}$$

to denote the class of languages accepted by automata of $\mathcal{M}$ with an oracle giving "advice" (a binary string fixed for any input length n) of length f(n).

Let $\mathcal{A}$ be a collection of alternating automata. Let $\mathcal{A}_{a(n)}$ be those automata of $\mathcal{A}$ which are a(n)-*alternation bounded* (where n is the input length).

3. Space Complexity of Probabilistic and Symmetric Alternating Automata

We first state some facts known about space bounded TMs, A-TMs, and R-TMs. We shall later use proof techniques similar to those used to establish these theorems.

THEOREM 3.1. [Savitch, 74] D-SPACE(S(n)) $\subseteq$ N-SPACE(S(n)) and for $S(n) \geq n$, N-SPACE(S(n)) $\subseteq$ D-SPACE($S(n)^2$).

THEOREM 3.2. [Chandra, Kozen, Stockmeyer, 78] For $S(n) \geq \log n$,

$$\text{A-SPACE}(S(n)) = \text{D-TIME}(2^{O(S(n))}).$$

THEOREM 3.3. (Due to Borodin in [Chandra, Kozen, and Stockmeyer,78]) For $S(n) \geq n$ and any alternation bound $a(n)$, $A_{a(n)}\text{-SPACE}(S(n)) \subseteq \text{D-SPACE}(S(n)(S(n)+a(n)))$.

THEOREM 3.4. [Gill, 77] For $S(n) \geq \log n$,

$$\text{R-SPACE}(S(n)) = \text{N-SPACE}(S(n)) .$$

This result by Gill is interesting, since his simulation of a S(n) space bounded N-TM by a S(n) space bounded R-TM require the R-TM to run in time $2^{2^{O(S(n))}}$. We *conjecture*:

$$\text{R-SC}^1(S(n)) \neq \text{NSPACE}(S(n))$$

in contrast to the *known fact*:

$$\text{N-SC}^1(S(n)) = \text{NSPACE}(S(n)).$$

(Note. To show this fact, just observe that any S(n) space bounded N-TM has time bound $\leq 2^{O(S(n))}$.)

THEOREM 3.5. For $S(n) \geq \log n$,

$$\text{RA-SPACE}(S(n)) = \text{A-SPACE}(S(n)) .$$

Proof. Since any RA-TM is an A-TM,

$$\text{RA-SPACE}(S(n)) \subseteq \text{A-SPACE}(S(n)).$$

To show the other containment, consider an A-TM, M with space bound $S(n) \geq \log n$. (S(n) is assumed to be constructable here, but this assumption can easily be dropped by the usual techniques, i.e., try $S(n) = 1,2,4,\ldots$. We assume constructability in the following proof as well.)

We construct M', a S(n)-space bounded RA-TM, as follows: At any given step of the simulation, M' will store two distinguished IDs of M

(1) the *current ID* I_C.

(2) the *home ID* I_H.

On input $w \in \Sigma^n$, M' sets both I_C, I_H to $\text{INIT}_M(w)$.

The simulation of M by M' will be partitioned into *phases* of *type* E (or U) during which M' explores only existential (or universal) IDs; M' is itself in a state type (E or U) corresponding to its phase type.

During each phase, M' executes a program (essentially that of [Gill, 77] to prove Theorem 3.4) which randomly chooses computation sequences from the home ID I_H (which is not modified during a phase). If at any time, M' reaches an ID, say J, of type different than its phase type, then M' alternates (i.e., switches from type E to U, or vice versa), sets $I_h \leftarrow J$, and thus enters a new phase.

By using a trick due to [Gill, 77] (i.e., choose a random $\gamma \in \{0,1\}^{2c \cdot S(n)}$ until finally $\gamma = \{0\}^{2c \cdot S(n)}$), M' can, by using only space $O(S(n))$, explore every relevant computation sequence (of the current phase type and starting at I_h) with high probability. Thus M' may simulate M. QED

The proof of Theorem 3.5 also implies:

COROLLARY. For any alternation bound $a(n)$,

$$AR_{a(n)} - SPACE(S(n)) = A_{a(n)} - SPACE(S(n)).$$

Thus $AR_{O(1)} - SPACE(S(n)) = A_{O(1)} - SPACE(S(n)) \subseteq D\text{-}SPACE(S(n)^2)$ by Theorem 3.3.

THEOREM 3.6. [Aleliunas, Karp, Lipton, Lovász and Rackoff, 79] and [Lewis and Papadimitriou, 80] For $S(n) \geq \log n$,

$$SYM - SPACE(S(n)) \subseteq R - SC^1(S(n)) .$$

This result was significant, since it relates space bounded SYM-TMs to simultaneously bounded R-TMs. We now generalize their results to alternating TMs.

THEOREM 3.7. For $S(n) \geq \log n$,

$$SYM - A - SPACE(S(n)) \subseteq RA - SC^1(S(n)) .$$

Proof. We prove this by induction on the alternation bound $a(n)$. The basis is from Theorem 3.6.

$$SYM\text{-}A_0\text{-}SPACE(S(n)) \subseteq RA_0\text{-}SC^1(S(n)) .$$

Let the induction hypothesis be

$$SYM\text{-}A_{a(n)}\text{-}SPACE(S(n)) \subseteq RA_{a(n)}\text{-}SC^1(S(n)) .$$

Then we apply Theorem 3.6 to establish

$$SYM\text{-}A_{a(n)+1}\text{-}SPACE(S(n)) \subseteq RA_{a(n)+1}\text{-}SC^1(S(n)) .$$ QED

4. Time Complexity of Probabilistic and Symmetric Alternating Automata

We first state some known results for time bounded alternating and probabilistic automata.

THEOREM 4.1. [Chandra, Kozen, Stockmeyer, 78] For $T(n) \geq n$,

$$\text{DSPACE}(T(n)) \geq \text{ATIME}(T(n))$$

and

$$\text{ATIME}(T(n)^2) \geq \text{NSPACE}(T(n)).$$

THEOREM 4.2. [Adleman, 78] For $T(n) \geq n$,

$$\text{R-TIME}(T(n)) \subseteq \text{DTIME}(T(n)^2)/O(T(n)^2)\text{advice}\ .$$

Adleman's proof here is quite elegant, showing that for any fixed $n \geq 0$, there is a set of $O(T(n))$ "witnesses" (each of length $O(T(n))$) which suffice to "derandomize" a R-TM on all inputs with length $\leq n$. We generalize his result to probabilistic alternating TMs.

THEOREM 4.3. For $T(n) \geq n$,

$$\text{RA-TIME}(T(n)) \leq \text{D-TIME}(T(n)^3)/O(T(n)^3)\text{advice}.$$

Proof. Note that a $T(n)$-time bounded RA-TM is also $T(n)$-alternation bounded. Then we repeatedly apply Adleman's proof technique for Theorem 4.2, using inductive hypothesis:

$$\text{RA}_{a(n)}\text{-TIME}(T(n)) \subseteq \text{DTIME}(T(n)^{2(a(n)+1)})/c \cdot T(n)^{2(a(n)+1)}\text{advice}$$

for some constant $c > 0$.

We turn our attention to time bounded symmetric computations.

THEOREM 4.4. [Lewis and Papadimitriou, 80]

$$\text{SYM-TIME}(T(n)) = \text{N-TIME}(T(n))\ .$$

THEOREM 4.5.

$$\text{SYM-A-TIME}(T(n)) = \text{A-TIME}(T(n))\ .$$

Proof. Since any SYM-A-TM is an A-TM, we need only prove

$$\text{SYM-A-TIME}(T(n)) \supseteq \text{A-TIME}(T(n))\ .$$

Let M be an A-TM with time bound $T(n)$, and input $w \in \Sigma^n$. The basic idea in the simulation (which was previously utilized by [Lewis and Papadimitriou, 80] for Theorem 4.4) is to separate the simulation into two *phases*:

(i) In the first phase, we play a symmetric game with the existential and universal players alternatively guessing a sequence of positive integers $i_1,\ldots,i_{T(n)}$ corresponding to branch choices down a path of the computation tree of M, from INIT(w) to a leaf.

(ii) In the next phase, we deterministically verify in time $T(n)$ the choice sequence is an accepting computation sequence.

Finally, we observe that deterministic computations can be made symmetric with no time increase by a technique of [Lewis and Papadimitriou, 80].

5. Computational Complexity of Probabilistic Parallel RAMs

[Fortune and Wyllie, 78] and [Savitch and Stimson, 78] have both proposed parallel RAM (PRAM) models derived from the RAM model of [Cook and Reckhow, 73]. Their PRAM models are similar, except [Fortune and Wyllie, 78] allow shared memory (where each location may be read simultaneously by many processes but written by only one process at a time). For brevity we adopt their PRAM model here; the reader should see their paper for the details (also see [Wyllie, 79]).

We let D-PRAM and N-PRAM denote respectively the deterministic and nondeterministic PRAM models defined by Fortune and Wyllie. We introduce here probabilistic PRAMs (denoted R-PRAMs) using the definition of Section 2.2 (note: these are *not* probabilistic alternating, as in Section 2.4).

THEOREM 5.1. [Fortune and Wyllie, 78] For $T(n) \geq \log n$,

$$\text{DSPACE}(T(n)^2) \supseteq \text{D-PRAM-TIME}(T(n)) \supseteq \text{DSPACE}(T(n)) .$$

THEOREM 5.2. For $T(n) \geq \log n$,

$$\text{R-SC}^2(T(n)) \supseteq \text{R-PRAM-TIME}(T(n)) \supseteq \text{R-SC}^1(T(n)) .$$

Proof. (Sketch) The first simulation (of a $T(n)$-time bound R-PRAM by a R-TM with simultaneous space $T(n)$ and time $2^{O(T(n))}$) is similar to Fortune and Wyllie's proof of the first containment of Theorem 5.1.

In the second simulation, we are given a R-TM, say M, with simultaneous space $T(n)$ and time $2^{O(T(n))}$. Our simulating R-PRAM M' has time $T(n)$ (assumed, for the moment, constructable). Given an input $w \in \Sigma^n$, M' essentially constructs in its shared memory a digraph D which is a subgraph of the computation graph G_M of M. Each ID of M of space $\leq T(n)$ will correspond to a node of D. However, D will have only one edge departing from each node v; this edge is randomly chosen from the set of edges departing v in the computation graph G_M. Then by definition of R-TMs (see Section 2.2), if M accepts w there is a path of D from INIT(w) to some ID in ACCEPT, with probability $\geq 1/2$, and if M rejects w, there is no such path for any D chosen.

To construct D, M' in time $O(T(n))$ initiates $2^{O(T(n))}$ new processes each, say P_i, with a distinct integer i representing $ID^{(i)} \in \mathcal{I}_M$. Then each P_i sets shared memory locations L[i] to some randomly chosen element of $NEXT_M(i)$. Then each processor sets L[i] to L[L[i]]. This is repeated synchronously T(n) times. If $ID^{(i_0)} = INIT(w)$, then at the end of this iteration,

$$L[i_0] \in ACCEPT \quad \text{iff} \quad M \quad \text{accepts} \quad w.$$

(This proof is similar to that of Fortune and Wyllie's proof of Theorem 5.1, except in their proof $G_M = D$, and thus no probabilistic choices need be taken).

THEOREM 5.3. For $T(n) \geq n$,

$$\text{R-RAM-TIME}(T(n)) \subseteq \text{D-RAM-TIME}(T(n)^2)/O(T(n)^2)\text{advice} .$$

Proof. Identical to Adleman's proof of Theorem 4.2.

THEOREM 5.4. For $T(n) \geq n$,

$$\text{R-PRAM-TIME}(T(n)) \subseteq \text{D-PRAM-TIME}(T(n)^3)/O(T(n)^3)\text{advice} .$$

Proof. Similar to Adleman's proof of Theorem 4.3.

The following is an analogue to Theorem 3.2 for space bounded parallel computations.

THEOREM 5.4. [Fortune and Wyllie, 78] For $S(n) \geq \log n$,

$$\text{N-PRAM-SPACE}(S(n)) = \text{N-TIME}(2^{O(S(n))}) .$$

THEOREM 5.5.

$$\text{R-PRAM-SPACE}(S(n)) = \text{N-PRAM}(S(n)) .$$

Proof is similar to that of Theorem 3.5.

(We also have similar results for probabilistic versions of the vector machines of [Pratt and Stockmeyer, 78].)

6. Computational Complexity of Probabilistic Circuits, Aggregates, and Hardware Modification Machines

We have various results, similar to the previous sections (and consistent with our probabilistic parallel processing thesis), circuits, aggregates of [Goldschlager, 78] and hardware modification machines of [Cook, 80]. Essentially, probabilistic circuits are identical to the uniform (with an appropriate definition of uniformity) circuits of [Borodin, 77], except they have special "random" nodes, labelled ? which output 0 ,1 with

equal probability (and independent of any other "random" nodes). Circuits are assumed acyclic, whereas aggregates may contain cycles.

The *hardware modification machines* (HMMs) of [Cook, 80] are similar to aggregates, except their *connections may change dynamically* as in the storage modification machines of [Schonhagen, 79] (HMMs also have complex input structures, which we ignore here). Cook introduced HMMs as a fundamental parallel processing model. We introduce *probabilistic* HMMs (R-HMMs) defined as in 2.2. For brevity, we only consider the time complexity of HMMs and R-HMMs.

THEOREM 6.1. [Dymond and Cook, 80]

$$\text{D-SPACE}(T(n)^2) \subseteq \text{HMM-TIME}(T(n)) \subseteq \text{D-SPACE}(T(n)) .$$

THEOREM 6.2.

$$\text{R-SC}^2(T(n)) \subseteq \text{R-HMM-TIME}(T(n)) \subseteq \text{R-SC}^1(T(n)) .$$

Proof. The first inclusion is found by simulating a T(n)-time bounded R-HMM-TM by a T(n)-time bounded R-PRAM and then applying Theorem 5.2. The second inclusion is found by a simulation to the proof of the second inclusion of Theorem 5.2; in this case the $2^{O(S(n))}$ hardware machines actually *form themselves into the digraph* D by randomly choosing a successor from NEXT. The shared memory instructions required in the previous proof are simulated by redirecting pointers appropriately, as in [Dymond and Cook, 80].

THEOREM 6.3. For $T(n) \geq n$,

$$\text{R-HMM-TIME}(T(n)) \subseteq \text{HMM-TIME}(T(n)^2)/O(T(n)^2)\text{advice} .$$

Proof is similar to Theorem 4.3.

REFERENCES

Adleman, L., "Two theorem on random polynomial time", *18th IEEE Symposium on Foundations of Computer Science*, 75-83, (1978).

Adleman, L., and Manders, K., "Reducibility, randomness, and intractibility", *Proc. 9th Annual ACM Symposium on Theory of Computing*, 151-163, (1977).

Aleliunas, R., R. M. Karp, R. H. Lipton, L. Lovasz and C. Rackoff, "Random walks, universal traversal sequences, and complexity of maze problems," *Proc. 20th Annual Symposium on Foundations of Computer Science*, 218-223 (1979).

Borodin, A., "On relating time and space to size and depth," *SIAM J. on Computing*, 6 (4), December 1977.

Chandra, A. K., D. C. Kozen and L. J. Stockmeyer, "Alternation", IBM Research Report, RC 7489 (1978).

Chandra, A. K. and L. J. Stockmeyer, "Alternation", *Conf. Record IEEE 17th Annual Symposium on Foundations of Computer Science*, 98-108,(1976).

Cook S. A., "Towards a complexity theory of synchronous parallel computation", Presented at *International Symposium uber Logik und Algorithmik zu Ehren von Professor Hort Specker*, Zurich, Switzerland, February 1980.

Cook, S. A., and R. A. Reckhow, "Time bounded random access machines", *J. Computer and Systems Sciences*, 7 (4): 354-375, 1973.

Dymond, P., and S. A. Cook, "Hardware complexity and parallel computation", *IEEE FOCS Conference*, 1980.

Dymond, P., Ph.D. Thesis, University of Toronto, Department of Computer Science, to appear.

Fortune, S., and J. Wyllie, "Parallelism in random access machines," In *Proc. of the 10th ACM Symposium on Theory of Computation*, 114-118,(1978).

Gill, J., "Complexity of probabilistic Turing machines", *SIAM J. of Computing*, 6 (4);675-695 (1977).

Goldschlager, L.,"A unified approach to models of synchronous parallel machines", In *Proc. 10th Annual ACM Symposium on the Theory of Computing*, San Diego, California, 89-94, (1978).

Goldschlager, L., "Synchronous parallel computation", Ph.D. Thesis and TR-114, Department of Computer Science, University of Toronto, December 1977.

Hong, J. W., "The similarity and duality of computation", *Conf. Record IEEE 21st Annual Symposium on Foundations of Computer Science*, 1980.

Karp, R. M., and R. J. Lipton, "Some connections between nonuniform and uniform complexity classes," *Proc. 17th Annual ACM Symposium on Theory of Computing*, April, 1980, 302-309. (Also presented at the Specker Symposium on Algorithms and Complexity, Zurich, February 1980.)

Kozen, D., "On parallelism in Turing Machines", *Conf. Record IEEE 17th Annual Symposium on Foundations of Computer Science*, 89-97 (1976).

Lewis, H. R., and C. H. Papadimitriou, "Symmetric space bounded computation", *ICALP80*.

Miller, C. L., "Rieman's hypothesis and tests for primality," Ph.D. Thesis, Berkeley (1975).

Pippenger, N., "On simultaneous resource bounds", In *Proc. 20th Annual Symposium on Foundations of Computer Science*, 307-211, (1979).

Pratt, V., and L. Stockmeyer, "A characterization of the powers of vector machines", *JCSS*, 12:198-211 (1978).

Rabin, M. O., "Probabilistic algorithms", *Algorithms and Complexity, New Directions and Recent Results*, edited by J. Traub, Academic Press.

Ruzzo, M. I., "On uniform circuit complexity", In *Proc. 20th Annual Symposium on Foundations of Computer Science*, 312-218, (1979).

Savitch, W. J., "Relationships between nondeterministic and deterministic tape complexities", *J. Computer and Systems Sciences*, 4(2):177-192.

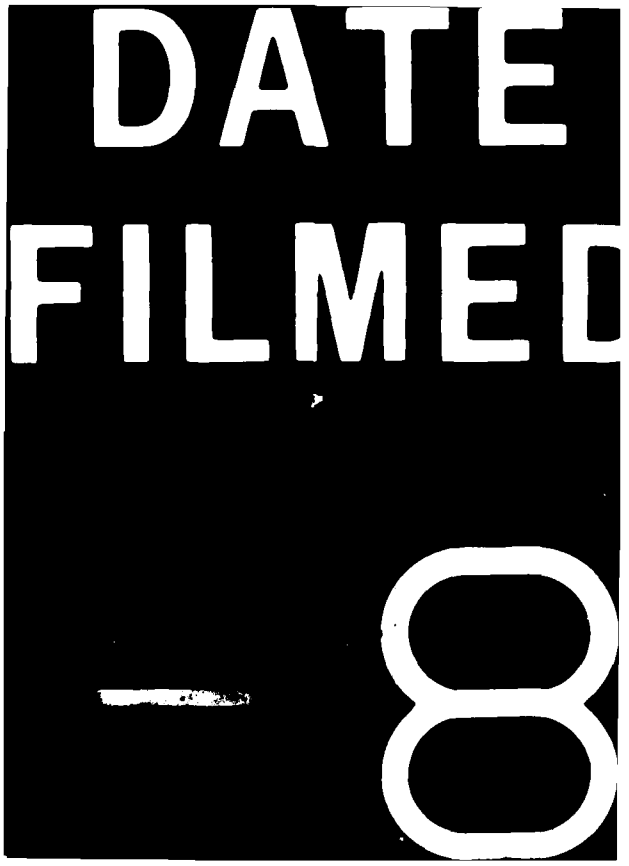
Savitch, W. J., and M. Stimson, "Time bounded random access machines with parallel processing", *JACM*, 26:103-118, January 1978.

Schonhage, A., Storage modification machines", Technical Report Mathematisches Institute, Universität Tubengen, Germany, 1979.

Strassen, V., and R. Solovay, "Fast Monte Carlo tests for primality", *SIAM J. Computing*.

Toffoli, T., "Reversible computing", MIT, Lab. for Computer Science, 1979.

Wyllie, J. C., "The complexity of parallel computations," PH.D. Thesis and TR-79-387, Dept. of Computer Science, Cornell University, 1979.


DATE
FILMED
8